IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Anil Seth et al.	Examiner: John J Romano
Serial No.:	10/087,296	Group Art Unit: 2192
Filed:	March 1, 2002	Docket No.: 1488.008US1
Title:	A TECHNIQUE FOR COMPILING COMPUTER CODE TO REDUCE ENERGY CONSUMPTION WHILE EXECUTING THE CODE	

SUPPLEMENTAL AMENDMENT & RESPONSE TO FINAL OFFICE ACTION UNDER 37 C.F.R. 1.116

Mail Stop RCE
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

        In response to the Advisory Action mailed <u>April 6, 2006,</u> please amend the application as follows:

IN THE CLAIMS

The claims are as follows.

1.   (Currently Amended) A method of compiling computer code including power-down instructions to reduce power consumption during execution of the code while satisfying user-specified real-time performance constraints on a microprocessor, comprising:

        identifying one or more potential locations in the computer code where the power-down instructions can be inserted;

        selecting locations to insert the power-down instructions from the identified potential locations in the code based on reducing power consumption and satisfying user-specified real-time performance constraints; and

        inserting the power-down instructions in the selected locations to reduce the power consumption during the execution of the code while satisfying user-specified real-time performance constraints.

2.   (Original) The method of claim 1, wherein the code is written for a microprocessor having distinct functional units.

3.   (Original) The method of claim 2, wherein identifying potential locations comprises:

        identifying potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

4.   (Original) The method of claim 3, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to executing the code.

AMENDMENT AND RESPONSE UNDER 37 CFR § 1.116 – EXPEDITED PROCEDURE   Page 3
Serial Number: 10/087,296   Dkt: 1488.008US1
Filing Date: March 1, 2002
Title: A TECHNIQUE FOR COMPILING COMPUTER CODE TO REDUCE ENERGY CONSUMPTION WHILE EXECUTING THE CODE

5.   (Original) The method of claim 4, wherein statically analyzing processor cycles is accomplished by statically analyzing the text in the code for the functional units not being used prior to executing the code.

6.   (Original) The method of claim 3, wherein each of the power-down instructions comprise:

    a first power-down instruction operable to reduce power to all of the at least one functional unit, such that the functional unit is placed in a low state of readiness and a second power-down instruction operable to reduce power to only a part of the at least one functional unit, such that the functional unit is placed in an intermediate state of readiness.

7.   (Currently Amended) The method of claim 1, wherein selecting identified potential locations on the computer code based on satisfying the user-specified real-time performance constraints, comprise:

    executing the code to generate power-profiling information associated with each of the identified potential locations;

    executing the code to generate execution path-profiling information associated with each of the identified potential locations;

    assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

    selecting the locations to insert the power-down instruction from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

8.   (Original) The method of claim 7, wherein executing the code to generate path-profiling information to each of the identified potential locations further comprises:

    generating execution probability of each of the identified potential locations based on the generated path-profiling information.

9.   (Original) The method of claim 8, wherein assigning the weight factor comprises:

      extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

      assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated execution probability.

10.   (Original) The method of claim 9, wherein assigning the weight factor further comprises:

      executing the code to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

      executing the code to assign a second weight factor based on execution probability at each of the identified potential locations;

      computing a product of the first and second weight factors for each of the identified potential locations;

      calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

      assigning the calculated weight factor to each of the identified potential locations.

11.   (Original) The method of claim 1, wherein user-specified real-time constraints comprise:

      the number of power-down instructions that can be inserted in an execution path, including one or more identified potential locations.

12.   (Currently Amended) The method of claim 11, wherein user-specified real-time performance constraints comprise:

      the number of additional cycles of execution time the user is willing to incur due to an insertion of the power-down instruction at each of the identified potential locations.

13.   (Original) The method of claim 11, further comprising:

inserting power-up instruction in the code to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

14.   (Currently Amended) A computer-readable medium having computer-executable instructions for reducing power consumption while running a computer program, comprising:

identifying one or more potential locations in the computer program where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the program based on satisfying user-specified real-time <u>performance</u> constraints; and

inserting the power-down instructions in the selected locations to reduce power consumption while running the computer program while satisfying the user-specified real-time <u>performance</u> constraints.

15.   (Original) The medium of claim 14, wherein the code is written for a microprocessor including distinct functional units.

16.   (Original) The medium of claim 14, wherein identifying potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

17.   (Original) The medium of claim 16, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to running the program.

18.   (Currently Amended) The medium of claim 14, wherein selecting the identified potential locations on the computer program based on satisfying the user-specified real-time constraints, comprise:

AMENDMENT AND RESPONSE UNDER 37 CFR § 1.116 – EXPEDITED PROCEDURE
Serial Number: 10/087,296
Filing Date: March 1, 2002
Title: A TECHNIQUE FOR COMPILING COMPUTER CODE TO REDUCE ENERGY CONSUMPTION WHILE EXECUTING THE
          CODE

Page 6
Dkt: 1488.008US1

running the computer program to generate power-profiling information associated with each of the identified potential locations;

running the computer program to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instructions from the identified locations based on the assigned weight factors and the user-specified real-time <u>performance</u> constraints.

19.   (Original) The medium of claim 18, wherein running the program to generate path-profiling information to each of the identified potential locations further comprises:

generating running probability of each of the identified potential locations based on the generated path-profiling information.

20.   (Original) The medium of claim 19, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated running probability.

21.   (Original) The medium of claim 20, wherein assigning the weight factor further comprises:

running the program to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

running the program to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

22.   (Currently Amended) The medium of claim 14, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in a running path including one or more identified potential locations.

23.   (Original) The medium of claim 22, further comprising:

inserting power-up instructions in the program to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

24.   (Currently Amended) A computer system for reducing power consumption during execution of computer code, comprising:

a storage device;

an output device; and

a processor programmed to repeatedly perform a method, comprising:

identifying one or more potential locations in the computer code where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the code based on satisfying user-specified real-time performance constraints; and

inserting the power-down instructions in the selected locations to reduce power consumption during the execution of the code while satisfying the user-specified real-time performance constraints.

25.   (Original) The system of claim 24, wherein the code is written for a microprocessor including distinct functional units.

26.   (Original) The system of claim 24, wherein identifying the potential locations comprises:

        identifying the potential locations based on the functional units not being used in the

potential locations, wherein the functional units not being used are determined based on

functional unit usage transfer functions at each of the potential locations as specified in standard

monotone data-flow frameworks.

27.   (Original) The system of claim 26, wherein identifying the potential locations is

accomplished by statically analyzing processor cycles prior to executing the code.

28.   (Currently Amended) The system of claim 24, wherein selecting the identified potential

locations on the computer code based on satisfying the user-specified real-time performance

constraints, comprises:

        executing the code to generate power-profiling information associated with each of the

identified potential locations;

        executing the code to generate execution path-profiling information associated with each

of the identified potential locations;

        assigning a weight factor to each of the identified potential locations based on the

generated power-profiling and path-profiling information; and

        selecting the locations to insert the power-down instruction from the identified locations

based on the assigned weight factors and the user-specified real-time performance constraints.

29.   (Original) The system of claim 28, wherein executing the code to generate path-profiling

information to each of the identified potential locations further comprises:

        generating execution probability of each of the identified potential locations based on the

generated path-profiling information.

30.   (Original) The system of claim 29, wherein assigning the weight factor comprises:

        extracting potential energy savings for each of the identified potential locations using the

generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated execution probability.

31.   (Original) The system of claim 30, wherein assigning the weight factor further comprises:

executing the code to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

executing the code to assign a second weight factor based on execution probability to each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

32.   (Currently Amended) The system of claim 24, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in an execution path including one or more identified potential locations.

33.   (Original) The system of claim 32, further comprising:

inserting power-up instructions in the code to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

34.   (Currently Amended) A computer-readable medium having a computer program including instructions for causing a computer to perform a method of selectively controlling power to different functional units of the computer, the instructions comprising:

power-down instructions inserted in the computer-program in selected locations based on reducing power consumption and satisfying user-specified real-time performance constraints; and

AMENDMENT AND RESPONSE UNDER 37 CFR § 1.116 – EXPEDITED PROCEDURE   **Page** 10
Serial Number: 10/087,296   Dkt: 1488.008US1
Filing Date: March 1, 2002
Title: A TECHNIQUE FOR COMPILING COMPUTER CODE TO REDUCE ENERGY CONSUMPTION WHILE EXECUTING THE
          CODE

wherein the power-down instruction in the selected locations reduce the power consumption during the execution of the code while satisfying the user-specified real-time <u>performance</u> constraints.

35.   (Currently Amended) The medium of claim 34, wherein inserting power-down instructions in the computer-program in selected locations further comprises:

identifying one or more potential locations in the computer program where power-down instructions can be inserted;

selecting locations to insert the power-down instructions from the identified potential locations in the program based on satisfying user-specified real-time <u>performance</u> constraints; and

inserting the power-down instructions in the selected locations to reduce power consumption while running the computer program while satisfying the user-specified real-time <u>performance</u> constraints.

36.   (Original) The medium of claim 35, wherein the code is written for a microprocessor including distinct functional units.

37.   (Original) The medium of claim 35, wherein identifying potential locations comprises:

identifying the potential locations based on the functional units not being used in the potential locations, wherein the functional units not being used are determined based on functional unit usage transfer functions at each of the potential locations as specified in standard monotone data-flow frameworks.

38.   (Original) The medium of claim 37, wherein identifying potential locations is accomplished by statically analyzing processor cycles prior to running the program.

39.   (Currently Amended) The medium of claim 35, wherein selecting the identified potential locations on the computer program based on satisfying the user-specified real-time constraints, comprise:

running the computer program to generate power-profiling information associated with each of the identified potential locations;

running the computer program to generate execution path-profiling information associated with each of the identified potential locations;

assigning a weight factor to each of the identified potential locations based on the generated power-profiling and path-profiling information; and

selecting the locations to insert the power-down instructions from the identified locations based on the assigned weight factors and the user-specified real-time performance constraints.

40.   (Original) The medium of claim 39, wherein running the program to generate path-profiling information to each of the identified potential locations further comprises:

generating running probability of each of the identified potential locations based on the generated path-profiling information.

41.   (Original) The medium of claim 40, wherein assigning the weight factor comprises:

extracting potential energy savings for each of the identified potential locations using the generated power profile analysis information; and

assigning the weight factor to each of the identified potential locations based on the extracted potential energy savings and the generated running probability.

42.   (Original) The medium of claim 41, wherein assigning the weight factor further comprises:

running the program to assign a first weight factor based on the extracted potential energy savings to each of the identified potential locations;

running the program to assign a second weight factor based on execution probability at each of the identified potential locations;

computing a product of the first and second weight factors for each of the identified potential locations;

calculating the weight factor for each of the identified potential locations based on the computed product of the first and second weight factors; and

assigning the calculated weight factor to each of the identified potential locations.

43.   (Currently Amended) The medium of claim 35, wherein user-specified real-time performance constraints comprise:

the number of power-down instructions that can be inserted in a running path including one or more identified potential locations.

44.   (Original) The medium of claim 43, further comprising:

inserting power-up instructions in the program to restore at least one functional unit to a ready state powered-down by the inserted power-down instructions.

REMARKS

This responds to the Office Action mailed on October 20, 2005.

Claims 1, 7, 12, 14, 18, 22, 24, 28, 32, 34, 35, 39 and 43 have been amended. Claims 1-44 are now pending in this application.

§103 Rejection of the Claims

Claims 1, 2, 11-15, 22-25, 32-36, 43 and 44 were rejected under 35 USC § 103(a) as being unpatentable over Bartley (U.S. Patent No. 6,219,796) in view of Y. Li et al. This rejection is respectfully traversed.

The present claims allow a tradeoff between performance, and power conservation based on user specified constraints for execution of program code. The references used to reject the claims either focus entirely on efficient use of power, or selection of sizes hardware to accomplish performance and energy usage goals. Neither reference, alone or combined teach or suggest the use of power down instructions in code to reduce power consumption while satisfying user specified performance constraints.

The claims have been amended to clearly point out reducing power consumption during the execution of the code while satisfying user-specified performance constraints to more clearly point out the difference between the claimed invention and the cited references.

The references will now be discussed in further detail to more clearly point out the differences between the references, alone or combined, and the claims.

Bartley describes scanning of code to determine that a functional unit of a processor will not be used during execution of a program. That functional unit is then shut down to conserve power. Instructions may be inserted to turn such functional units off and on. Thresholds may be used to make sure that at least some benefit is provided by doing so. As noted several times in Col. 7, Bartley focuses on ensuring efficient use of power. Lines 17-18, and 38. The entire context of Col. 7 is to efficiently use power. Despite references to static and dynamic program analysis, and thresholds, they all relate to identifying sections of code where a shutdown may

occur to efficiently use power, and do not relate to code performance. No mention is made of code performance in Bartley, only efficient use of power.

Y. Li et al. is used to modify hardware, such as cache and main memory size to trade off performance and energy use goals. It has nothing to do with inserting power down instructions in code. It is a very inflexible application of power conservation techniques to a dedicated hardware platform. Once done, it is optimized to an individual application. Thus, it teaches away from the ability, or even any desire to use power control instructions in programs. Neither reference, alone or combined teaches or suggests inserting power down instructions to reduce power consumption while satisfying user-specified real-time constraints.

The references are also not properly combinable, as no proper motivation to combine them has been provided, and is it not clear how they can be combined to arrive at the claimed invention.

Applicant believes that the current amendments to the claims overcomes all the prior rejections of the claims, and reserves the right provide arguments to such rejections should they be repeated in subsequent actions.

Claims 3-10, 16-21, 26-31 and 37-42 were rejected under 35 USC § 103(a) as being unpatentable over Bartley (U.S. Patent No. 6,219,796) in view of Y. Li et al. ("A Framework for Estimating and Minimizing Energy Dissipation of Embedded HW/SW Systems", Proceedings of the 35th Design Automation Conference, (1998), p. 188-193), and further in view of G. Ramalingam ("Data Flow Frequency Analysis", SIGPLAN Conference on Programming Language Design and Implementation, 1996). This rejection is respectfully traversed, as all such rejected claims depend from independent claims that are believed allowable.

CONCLUSION

Applicant respectfully submits that the claims are in condition for allowance and notification to that effect is earnestly requested. The Examiner is invited to telephone Applicant's attorney (612) 373-6972 to facilitate prosecution of this application.

If necessary, please charge any additional fees or credit overpayment to Deposit Account No. 19-0743.
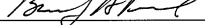
Respectfully submitted,

ANIL SETH ET AL.

By their Representatives,

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. Box 2938
Minneapolis, MN 55402
(612) 373-6972

Date 4-19-2006   By _____
                            Bradley A. Forrest
                            Reg. No. 30,837